Fitness-AUC Bandit Adaptive Strategy Selection vs. the Probability Matching one within Differential Evolution: An Empirical Comparison on the BBOB-2010 Noiseless Testbed

Álvaro Fialho, Marc Schoenauer, Michèle Sebag

Fitness-AUC Bandit Adaptive Strategy Selection vs. the Probability Matching one within Differential Evolution

An Empirical Comparison on the BBOB-2010 Noiseless Testbed

Álvaro Fialho[1], Marc Schoenauer[1,2], Michèle Sebag[1,2]
[1]Microsoft Research–INRIA Joint Centre
Parc Orsay Université
91893 Orsay Cedex, France
[2]Project-Team TAO, INRIA Saclay - Île-de-France
LRI, Bât. 490, Université Paris-Sud
91405 Orsay Cedex, France
FirstName.LastName@inria.fr

ABSTRACT

The choice of which of the available strategies should be used within the Differential Evolution algorithm for a given problem is not trivial, besides being problem-dependent and very sensitive with relation to the algorithm performance. This decision can be made in an autonomous way, by the use of the *Adaptive Strategy Selection* paradigm, that continuously selects which strategy should be used for the next offspring generation, based on the performance achieved by each of the available ones on the current optimization process, *i.e.*, while solving the problem. In this paper, we use the BBOB-2010 noiseless benchmarking suite to better empirically validate a comparison-based technique recently proposed to do so, the Fitness-based Area-Under-Curve Bandit [4], referred to as *F-AUC-Bandit*. It is compared with another recently proposed approach that uses Probability Matching technique based on the relative fitness improvements, referred to as *PM-AdapSS-DE* [7].

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*global optimization, unconstrained optimization*; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems; I.2.8 [**Computing Methodologies**]: Artificial Intelligence: Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Benchmarking, Black-box optimization, Adaptive Strategy Selection, Comparison-based, ROC Area Under Curve, Multi-Armed Bandits

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

1. INTRODUCTION

Differential Evolution (DE) is a very popular evolutionary algorithm, mainly because of its simple structure, ease of use, robustness and speed. Because of this, DE has been applied on many real-world applications, such as pattern recognition, neural network training, data mining, etc.

However, one of the features that helps making it robust with relation to so many different situations – the number of available strategies for offspring generation [16, 14] – is also the responsible for adding an extra difficulty to its use, the definition of which of the available strategies should be applied to the problem at hand. Such choice is problem-dependent, and very sensitive in terms of algorithm performance, what turns to be a non-trivial decision for the user.

An off-line tuning procedure might be used to find the best strategy for the problem at hand. But, besides being computationally expensive, its result (the best single strategy) will always lead to sub-optimal behavior, as exploration tends to be more important in the beginning of the optimization process, while exploitation should be preferred when approaching to the optimum. In other words, different strategies should be applied at different moments of the optimization process, according to its "current needs" in terms of exploration and exploitation.

Defining the way such mixture of strategies will be used during the process becomes yet another optimization problem. This is the main motivation for the use of *Adaptive Strategy Selection* techniques: based on the recent performance of each strategy on the current optimization process, the strategy to be used on the generation of the next offspring is automatically chosen, while solving the problem.

A new comparison-based technique, the *F-AUC-Bandit*, has been recently proposed to this aim [4], being originally assessed in the context of adaptive operator selection within Genetic Algorithms (GAs). It uses a multi-armed bandit algorithm to select the strategy to be applied, with the Area Under the ROC Curve paradigm [3] being used to assess the performance of each strategy, based on the ranks of the fitnesses of the generated offspring, what makes it totally invariant with relation to monotonous transformations over the fitness function.

In this paper we extend its empirical validation, coupling it with a DE algorithm, and analyzing it on the light of the BBOB-2010 noiseless benchmarking suite. Another recently proposed technique, the *PM-AdapSS-DE* [7], is used

as baseline for comparison. It has a much simpler scheme, although providing competitive results.

Both techniques are briefly described in Section 2 (we refer the reader to the original papers [4, 7] for a more complete view). Section 3 presents the experimental settings that were used to generate the results presented in Section 4. Finally, Section 5 presents the CPU timing experiments, and the paper is concluded with some final considerations in Section 6.

2. ALGORITHMS PRESENTATION

To do *Adaptive Strategy* (or Operator) *Selection*, there is the need of defining how the impact of the application of a given strategy is assessed, *i.e.*, how to reward the strategy after its "production", what is referred to as the *Credit Assignment* mechanism; and based on these assessments, there is the need of defining how to select the next strategy to be applied, what is called as the *Strategy Selection* scheme. Both the techniques being analyzed in this paper will be briefly described in the following, focusing on how they handle these two issues.

2.1 The Fitness-AUC Bandit algorithm

The *F-AUC-Bandit* algorithm, recently proposed in the context of GAs [4], uses the Area Under the ROC Curve (AUC) paradigm to assess the empirical quality of each strategy. The AUC is a criterion originally used in *Machine Learning* to compare binary classification rules [3]. In the context of *Adaptive Strategy Selection*, it shows how good one strategy is, by comparing the rewards received after its recent applications with the rewards received by the others.

Instead of being calculated based on the received raw rewards, in this work the AUC uses the ranks of these rewards, what improves its robustness with relation to different problems (there is no need of re-scaling the algorithm to the different possible ranges of rewards). Besides, by directly using the rank of the fitnesses of the generated offspring, instead of the commonly used fitness improvements, it becomes a total comparison-based method, invariant with relation to monotonous transformations over the fitness function. This is what we refer to as the *Fitness-based AUC* credit assignment scheme. In case the generated offspring does not improve over its parent, a null reward is assigned.

Figure 1, reproduced from [4], illustrates an example computation of the AUC. Briefly, it is the total area upper bounded by the *Receiving Operator Curve* (ROC), represented by the solid line in the example. Computing the quality of a given strategy consists of going down the sorted list of raw rewards, and drawing, starting from the origin, a vertical segment each time the strategy under assessment is found in the list, a horizontal one otherwise, and a diagonal in case of ties.

In this example, for the sake of clarity, each rank position has the same weight on the calculation of the reward, *i.e.*, each segment has the same length than the others, no matter its ranking. But it is clear that the initial rank positions (the best raw rewards) should have a higher impact on this quality estimation. To this aim, a decay factor can be applied. Being W the size of the sliding window that stores the recent raw rewards received by all the strategies, and r the rank position of a given reward, the length of its segment in the ROC curve (*i.e.*, its importance in the AUC computation) can be calculated as $D^r(W - r)$, with

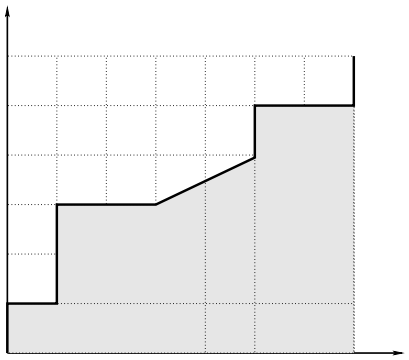

Figure 1: Sample computation of AUC reward: only two operators are involved, and the sorted list contains the operators in the order (1 2 1 1 2 2 [2 2 1] 1 2 2 1), with [2 2 1] meaning that these 3 positions have the same raw reward, leading to the diagonal line between points (3 3) and (5 4) (dotted lines are spaced by 1). In case of decay, the width of the squares would decrease leftward and upward.

$D \in]0, 1]$ being the decay factor that defines how skewed is this ranking distribution. A linear decay is achieved by using $D = 1$; smaller D, faster the decay.

A multi-armed bandit technique, based on the UCB multi-armed bandit formula [1, 4], is then used to select the next strategy to be applied, according to the presented quality estimation. The main difference is that, as there is no much sense in calculating statistics over statistics, in this case the empirical quality used by the UCB formula is equal to the last received reward (the AUC computation), instead of being an average of the received rewards, since the AUC efficiently summarizes the up-to-date quality of the strategy with relation to the others.

Besides the decay factor D, the *F-AUC-Bandit* algorithm requires the definition of a scaling factor C, that is used to balance the importance of its exploration and exploitation terms; and also the credits sliding window size W.

2.2 The PM-AdapSS-DE algorithm

The *PM-AdapSS-DE* algorithm is much simpler, although presenting competitive results, as shown in its original paper [7]. The credit assignment scheme awards the strategy with the absolute average of the raw rewards recently received by it. The raw reward in this case is the relative fitness improvement (the improvement achieved by the offspring over its parent, normalized by the fitness of the best-so-far individual).

The Probability Matching (PM) technique [6] uses this received credit to update the empirical quality estimate it keeps for each strategy, with the weight of the received reward being ruled by a user-defined parameter, the adaptation rate $\alpha \in]0, 1]$. The probability of selection of each strategy is then updated proportionally to its empirical estimate with relation to the others. A minimal probability $p_{min} \in [0, 1]$ might be applied, so that no operator gets "lost" during the process [17]. With this, every time a strategy needs to be applied, it is selected from the set of available ones by a roulette-wheel-like process over these up-to-date probabilities.

3. EXPERIMENTAL SETTINGS

The results presented in Sections 4 and 5 represent the performance of the adaptive schemes at their best, *i.e.*, using the best configuration found for their user-defined parameters. As in [4], the F-Race technique [2] was used for the off-line tuning, starting to eliminate the configurations after 1 run over all instances of a given dimension (totalizing 720 instances), with the Friedman's two-way analysis of variances by ranks statistical test being applied at a rate of 95%, up to 10 runs or just one configuration left. For the *F-AUC-Bandit*, the following parameter values were tried: scaling factor $C \in \{.1, .5, 1, 5, 10\}$, decay factor $D \in \{.5, 1\}$, and window size $W \in \{50, 100, 500\}$, summing up to 30 configurations. And for the *PM-AdapSS-DE*, the minimal probability $p_{min} \in \{0, .05, .1, .2\}$, and the adaptation rate $\alpha \in \{.1, .3, .6, .9\}$ were tried, totalizing 16 configurations.

Although different tuning procedures were executed independently for each dimension $\in \{5, 20\}$ (a kind of representative set of all the analyzed dimensions), the same configuration was found to be the best for both dimensions, for both techniques. This confirms the already mentioned robustness gain provided by the use of relative rewards instead of raw ones: the parameters of the adaptive scheme do not need to be re-scaled for each problem, a same parameter configuration might be well-performing on an entire class of problems. In the case of the *F-AUC-Bandit*, given its comparison-based property, this is ensured to be true for all the class of fitness functions defined by monotonous transformations over the original one. After this tuning phase, the *F-AUC-Bandit* was set up with the following parameter values: $\{D = .5, C = .5, W = 50\}$, and the *PM-AdapSS-DE* was assessed with $\{p_{min} = 0, \alpha = .6\}$.

Both techniques were coupled to a standard Differential Evolution algorithm. Given that the main focus of this work it rather to further empirically assess the *Adaptive Strategy Selection* techniques, instead of competing with the best optimizers, it is true to say that no much attention was deserved to the user-defined parameters of DE. The population size NP was fixed at 10 times the dimension, and the mutation scaling factor F was set to 0.5. Although a crossover rate CR of 0.9 is usually advocated, it was chosen to set it to 1.0 for this benchmarking exercise, as in this way the DE becomes invariant with relation to rotation, and entirely dependent on the application of the mutation strategies [10].

As in [7], the *Adaptive Strategy Selection* techniques had as objective to efficiently select, while solving the problem, between the following set of strategies:

1) "DE/rand/1": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3})$
2) "DE/rand/2": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5})$
3) "DE/rand-to-best/2": $\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{best} - \mathbf{x}_{r_1}) + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5})$
4) "DE/current-to-rand/1": $\mathbf{v}_i = \mathbf{x}_i + F \cdot (\mathbf{x}_{r_1} - \mathbf{x}_i) + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3})$

where $\mathbf{x}_i$ represents the current individual, $\mathbf{x}_{best}$ is the best individual in the current generation, r_1, r_2, r_3, r_4, r_5 are individuals randomly chosen from the population, being $r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq i$. F is the mutation scaling factor.

The final experiments were performed following the BBOB guidelines [8], with the maximum number of evaluations being fixed at 10^5 times the dimension. The mentioned parameter values were used on all the experiments, for all dimensions, thus the crafting effort for both techniques is equal to zero.

4. RESULTS

Results from experiments according to [8] on the benchmark functions given in [5, 9] are presented in Figures 2, 3 and 4 and in Table 1. The **expected running time (ERT)**, used in the figures and table, depends on a given target function value, $f_t = f_{opt} + \Delta f$, and is computed over all relevant trials as the number of function evaluations executed during each trial while the best function value did not reach f_t, summed over all trials and divided by the number of trials that actually reached f_t [8, 13]. **Statistical significance** is tested with the rank-sum test for a given target Δf_t (10^{-8} in Figure 2) using, for each trial, either the number of needed function evaluations to reach Δf_t (inverted and multiplied by -1), or, if the target was not reached, the best Δf-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration.

5. CPU TIMING EXPERIMENTS

For the timing experiments, both algorithms were run on f8 and restarted until at least 30 seconds (according to [8]). The experiments have been conducted with an Intel Xeon E5345 processor (2.33 GHz) running Linux 2.6.31.12. The same C++ implementation (gcc version 4.4.1) was used for both, with the only difference being the portions of code referring to the strategy selection. For the *F-AUC-Bandit*, the results were 6.3, 7.9, 8.0, 8.4, 8.8, 9.1 $\times 10^{-6}$ seconds per function evaluation, for the dimensions 2, 3, 5, 10, 20, 40 respectively. For the baseline technique, the *PM-AdapSS-DE*, the results were 1.6, 1.6, 1.6, 1.8, 2.1 and 2.9 $\times 10^{-6}$ seconds per function evaluation, for the dimensions 2, 3, 5, 10, 20, 40 respectively. The timing difference between both is mostly explained by their respective credit assignment schemes: the latter uses a simple mechanism (average of rewards), while the former has a much more complex and time-consuming one (area under curve), with the advantage of being a comparison-based method.

For the sake of further comparison, the same Differential Evolution code, but with a uniform strategy selection instead of a "laborious" one, presents as results 3.3, 3.5 4.0, 5.5, 8.3 and 16 $\times 10^{-7}$ seconds per function evaluation, for the dimensions 2, 3, 5, 10, 20, 40 respectively. The difference between these results and the ones achieved by the adaptive schemes shows exactly the price to be paid for the use of them.

6. FINAL CONSIDERATIONS

This paper presented a more detailed empirical validation of a recently proposed *Adaptive Strategy Selection* technique, the *F-AUC-Bandit* [4]. It uses the multi-armed bandit approach to select between the available strategies while solving the problem, based on an empirical quality estimation assessed by means of the Area Under the Curve (AUC) paradigm. The AUC is computed over the ranks of the fitnesses of the generated offspring, what provides to this technique the comparison-based characteristic, *i.e.*, it turns

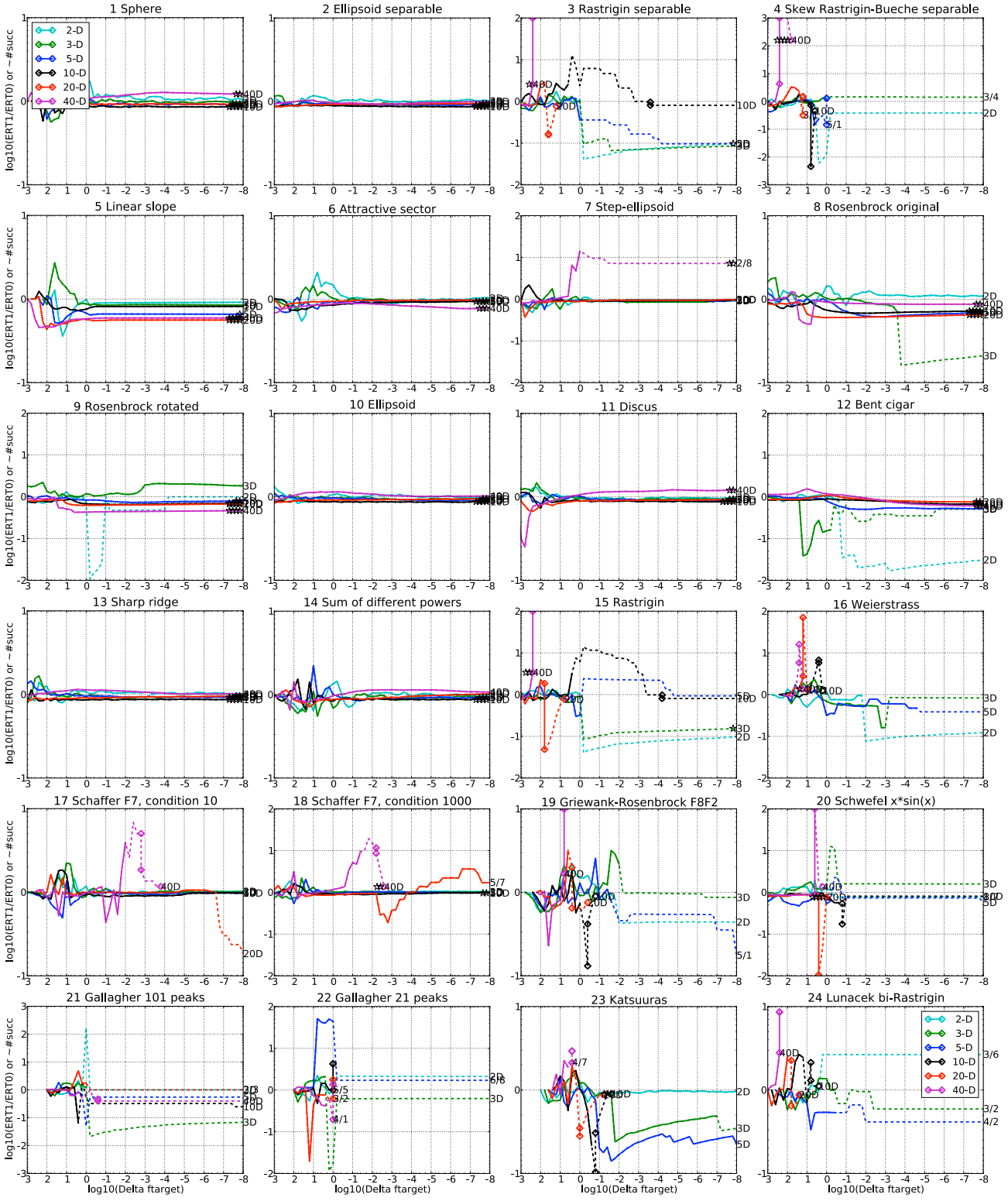

Figure 2: ERT ratio of F-AUC-Bandit divided by PM-AdapSS-DE versus $\log_{10}(\Delta f)$ for f_1–f_{24} in 2, 3, 5, 10, 20, 40-D. Ratios $< 10^0$ indicate an advantage of F-AUC-Bandit, smaller values are always better. The line gets dashed when for any algorithm the ERT exceeds thrice the median of the trial-wise overall number of f-evaluations for the same algorithm on this function. Symbols indicate the best achieved Δf-value of one algorithm (ERT gets undefined to the right). The dashed line continues as the fraction of successful trials of the other algorithm, where 0 means 0% and the y-axis limits mean 100%, values below zero for F-AUC-Bandit. The line ends when no algorithm reaches Δf anymore. The number of successful trials is given, only if it was in $\{1 \dots 9\}$ for F-AUC-Bandit (1st number) and non-zero for PM-AdapSS-DE (2nd number). Results are significant with $p = 0.05$ for one star and $p = 10^{-\#\star}$ otherwise, with Bonferroni correction within each figure.

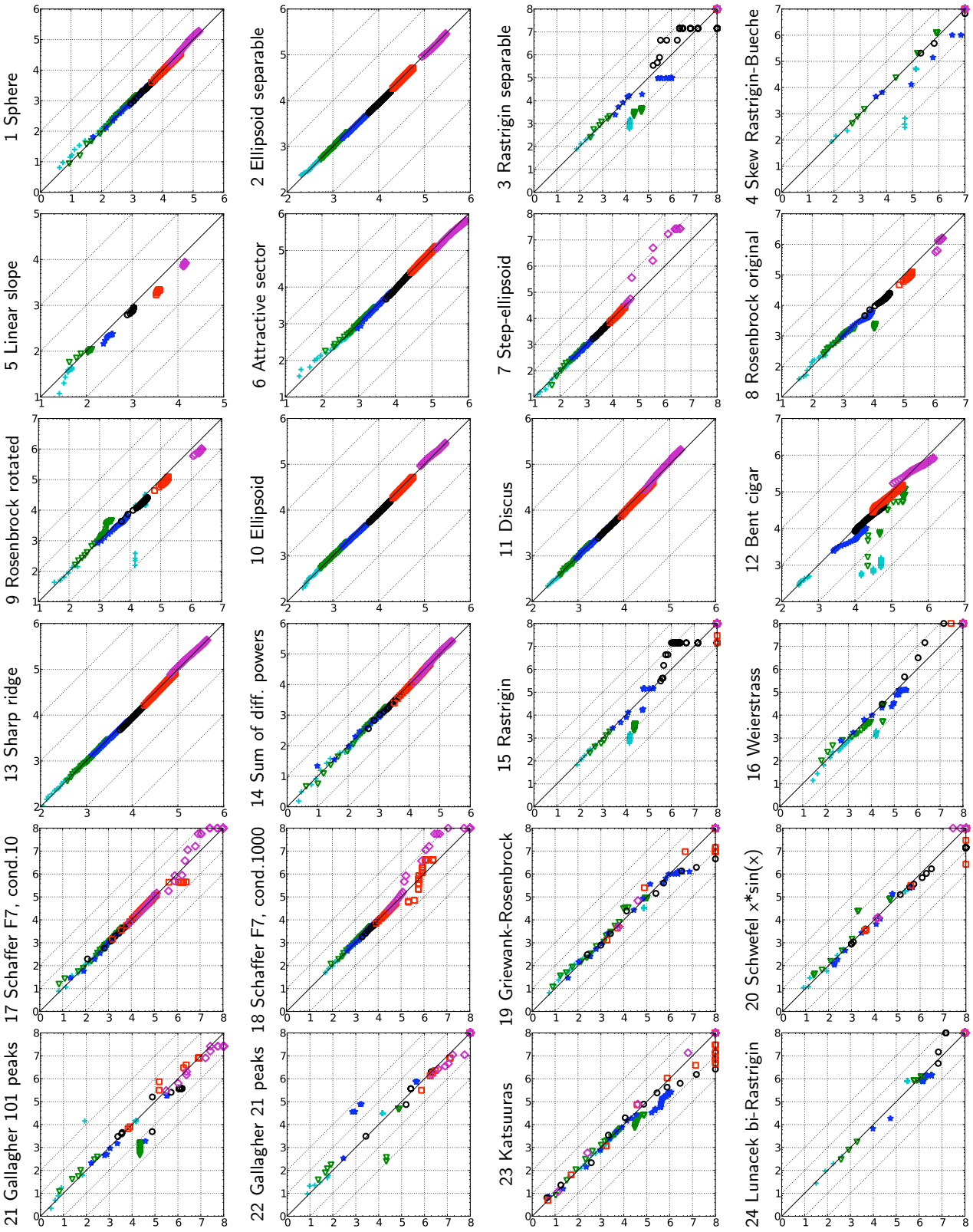

Figure 3: Expected running time (ERT in log10 of number of function evaluations) of F-AUC-Bandit versus PM-AdapSS-DE for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_1–f_{24}. Markers on the upper or right egde indicate that the target value was never reached by F-AUC-Bandit or PM-AdapSS-DE respectively. Markers represent dimension: 2:+, 3:▽, 5:⋆, 10:○, 20:□, 40:◇.

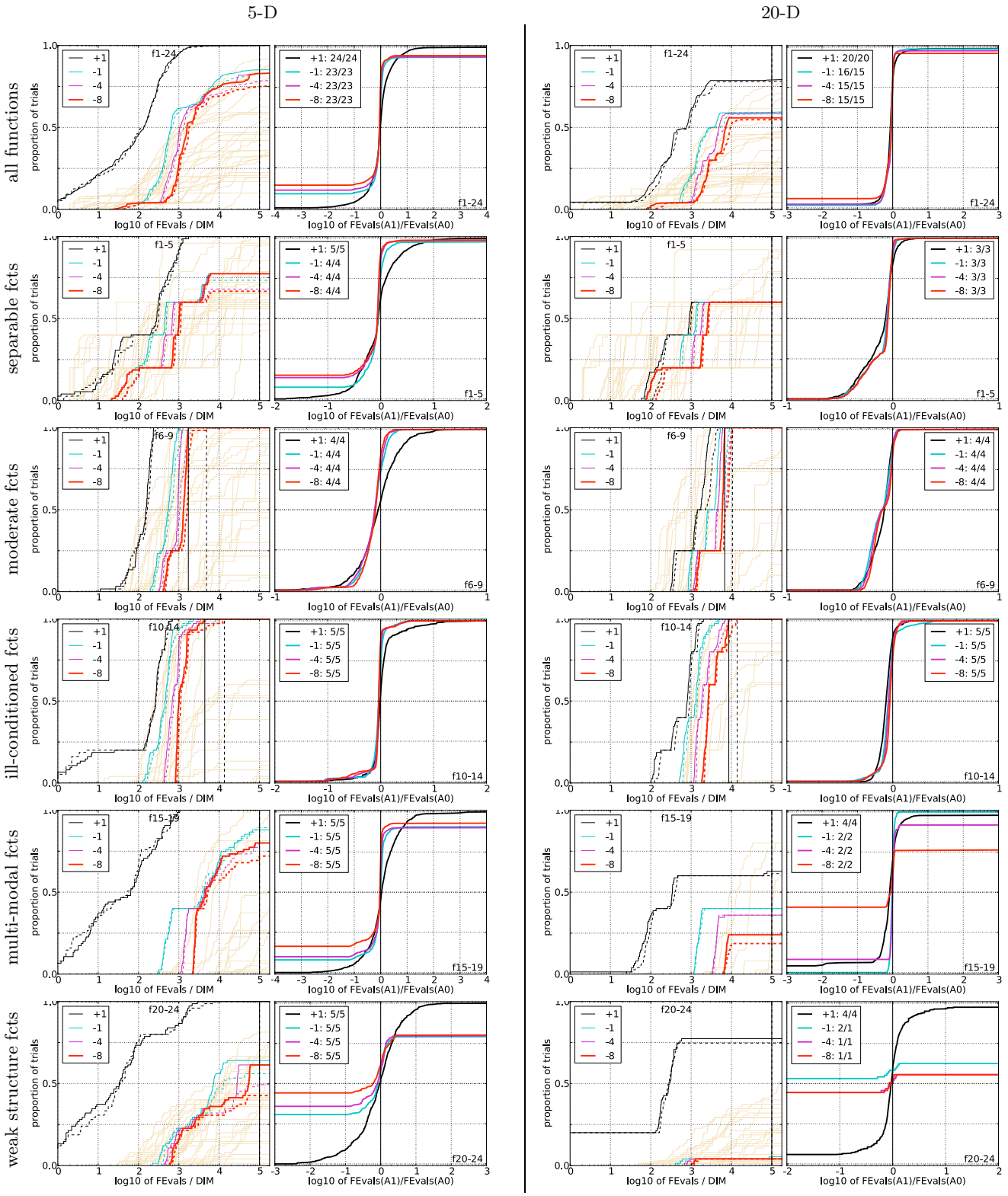

Figure 4: Empirical cumulative distributions (ECDF) of run lengths and speed-up ratios in 5-D (left) and 20-D (right). Left sub-columns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\mathrm{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for F-AUC-Bandit (solid) and PM-AdapSS-DE (dashed). Light beige lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right sub-columns: ECDF of FEval ratios of F-AUC-Bandit divided by PM-AdapSS-DE, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being > 0 or < 1. The legends indicate the number of functions that were solved in at least one trial (F-AUC-Bandit first).

5-D

Δf	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
f_1	11	12	12	12	12	12	15/15
0: AdapSS	4.8	39	74	140	220	290	15/15
1: Bandit	5.9	37	67	130	**200**$^{\star 2}$	**270**$^{\star 2}$	15/15
f_2	83	87	88	90	92	94	15/15
0: AdapSS	19	24	28	38	47	55	15/15
1: Bandit	18	**22***	**26**$^{\star 2}$	**35**$^{\star 2}$	**42**$^{\star 2}$	**50**$^{\star 3}$	15/15
f_3	720	1600	1600	1600	1700	1700	15/15
0: AdapSS	4.9	33	170	360	620	620	5/15
1: Bandit	3.4	12	59	60	60	60	13/15
f_4	810	1600	1700	1800	1900	1900	15/15
0: AdapSS	4.9	4.3e3	∞	∞	∞	∞*5.0e5*	0/15
1: Bandit	5.7	620	∞	∞	∞	∞*5.0e5*	0/15
f_5	10	10	10	10	10	10	15/15
0: AdapSS	24	33	36	36	36	36	15/15
1: Bandit	15	23	24	24	24	24	15/15
f_6	110	210	280	580	1000	1300	15/15
0: AdapSS	8.1	8.1	9	7.3	5.7	5.7	15/15
1: Bandit	6.5	7.6	8.5	7	5.3	5.4	15/15
f_7	24	320	1200	1600	1600	1600	15/15
0: AdapSS	12	2.5	1.2	1.3	1.3	1.5	15/15
1: Bandit	13	2.4	1.1	1.2	1.2	1.3	15/15
f_8	73	270	340	390	410	420	15/15
0: AdapSS	13	9.4	15	19	20	21	15/15
1: Bandit	13	8.9	11	**11**$^{\star 2}$	**13**$^{\star 2}$	**14**$^{\star 2}$	15/15
f_9	35	130	210	300	340	370	15/15
0: AdapSS	25	22	21	21	21	22	15/15
1: Bandit	24	18	16	**16***	**16***	**17***	15/15
f_{10}	350	500	570	630	830	880	15/15
0: AdapSS	4.6	4	4.3	5.3	5.2	5.9	15/15
1: Bandit	4.5	3.9	4	4.9	**4.7***	**5.3**$^{\star 2}$	15/15
f_{11}	140	200	760	1200	1500	1700	15/15
0: AdapSS	6.3	6.5	2.3	2.3	2.4	2.7	15/15
1: Bandit	6.2	6.6	2.2	**2**$^{\star 2}$	**2.2***	**2.4**$^{\star 3}$	15/15
f_{12}	110	270	370	460	1300	1500	15/15
0: AdapSS	24	16	24	27	12	13	15/15
1: Bandit	22	13	13	14	6.4	6.6	15/15
f_{13}	130	190	250	1300	1800	2300	15/15
0: AdapSS	9.9	11	12	3.4	3.5	3.4	15/15
1: Bandit	10	11	11	**3.2**$^{\star 2}$	**3.2**$^{\star 2}$	**3.1**$^{\star 2}$	15/15
f_{14}	9.8	41	58	140	250	480	15/15
0: AdapSS	1	9.4	17	15	12	8.9	15/15
1: Bandit	2.2	9.4	15	**13***	**11***	**8.2***	15/15
f_{15}	510	9300	1.9e4	2.0e4	2.1e4	2.1e4	14/15
0: AdapSS	5.4	6.3	3.2	3.2	7.6	7.4	12/15
1: Bandit	5.4	2	7.5	7.3	7.1	6.9	12/15
f_{16}	120	610	2700	1.0e4	1.2e4	1.2e4	15/15
0: AdapSS	3.9	120	55	20	28	27	11/15
1: Bandit	6.4	39	31	12	11	10	13/15
f_{17}	5.2	210	900	3700	6400	7900	15/15
0: AdapSS	4.2	4	2.3	1.4	1.3	1.4	15/15
1: Bandit	5.7	3.9	2.3	1.3	1.3	1.4	15/15
f_{18}	100	380	4000	9300	1.1e4	1.2e4	15/15
0: AdapSS	4.3	4.1	0.76	0.67	0.89	1	15/15
1: Bandit	4.3	3.8	0.75	0.69	0.88	0.98	15/15
f_{19}	1	1	240	1.2e5	1.2e5	1.2e5	15/15
0: AdapSS	37	2.0e3	2.1e3	19	19	29	1/15
1: Bandit	29	2.6e3	1.7e3	11	11	10	5/15
f_{20}	16	850	3.8e4	5.4e4	5.5e4	5.5e4	14/15
0: AdapSS	11	14	9.4	6.6	6.6	6.6	8/15
1: Bandit	6.8	7.6	6.9	4.9	4.8	4.8	10/15
f_{21}	41	1200	1700	1700	1700	1800	14/15
0: AdapSS	4	33	200	200	200	190	9/15
1: Bandit	5.2	1.7	110	110	110	110	11/15
f_{22}	71	390	940	1000	1000	1100	14/15
0: AdapSS	4.1	4.6	470	440	420	410	8/15
1: Bandit	4.8	200	800	750	720	710	6/15
f_{23}	3	520	1.4e4	3.2e4	3.3e4	3.4e4	15/15
0: AdapSS	1.8	8.9	12	14	21	27	5/15
1: Bandit	2.4	9.3	2.3	3.4	5.3	6.9	15/15
f_{24}	1600	2.2e5	6.4e6	9.6e6	1.3e7	1.3e7	3/15
0: AdapSS	5.9	6.8	0.24	0.36	0.27	0.27	2/15
1: Bandit	4.1	3.7	0.13	0.15	0.11	0.11	4/15

20-D

Δf	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
f_1	43	43	43	43	43	43	15/15
0: AdapSS	100	200	290	470	650	830	15/15
1: Bandit	93	180	**260***	430	**600***	**760**$^{\star 2}$	15/15
f_2	380	390	390	390	390	390	15/15
0: AdapSS	52	63	73	93	110	130	15/15
1: Bandit	**48***	**58***	**68***	**86**$^{\star 2}$	**100**$^{\star 2}$	**120***	15/15
f_3	5100	7600	7600	7600	7600	7700	15/15
0: AdapSS	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
f_4	4700	7600	7700	7700	7800	1.4e5	9/15
0: AdapSS	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
f_5	41	41	41	41	41	41	15/15
0: AdapSS	82	92	96	96	96	96	15/15
1: Bandit	**42**$^{\star 3}$	**52**$^{\star 2}$	**53**$^{\star 3}$	**54**$^{\star 3}$	**54**$^{\star 3}$	**54**$^{\star 3}$	15/15
f_6	1300	2300	3400	5200	6700	8400	15/15
0: AdapSS	20	16	15	14	14	14	15/15
1: Bandit	19	15	14	14	14	14	15/15
f_7	1400	4300	9500	1.7e4	1.7e4	1.7e4	15/15
0: AdapSS	5.9	3.4	2.1	1.6	1.6	1.7	15/15
1: Bandit	**5.2**$^{\star 2}$	**3.1***	1.9	1.6	1.6	1.6	15/15
f_8	2000	3900	4000	4200	4400	4500	15/15
0: AdapSS	35	35	38	39	40	41	15/15
1: Bandit	**23**$^{\star 3}$	**21**$^{\star 3}$	**23**$^{\star 3}$	**24**$^{\star 3}$	**25**$^{\star 3}$	**26**$^{\star 3}$	15/15
f_9	1700	3100	3300	3500	3600	3700	15/15
0: AdapSS	37	41	43	45	46	47	15/15
1: Bandit	**26**$^{\star 3}$	**25**$^{\star 3}$	**27**$^{\star 3}$	**29**$^{\star 3}$	**30**$^{\star 3}$	**31**$^{\star 3}$	15/15
f_{10}	7400	8700	1.1e4	1.5e4	1.7e4	1.7e4	15/15
0: AdapSS	2.7	2.8	2.6	2.4	2.6	3	15/15
1: Bandit	**2.5***	**2.6**$^{\star 2}$	**2.4**$^{\star 2}$	**2.2**$^{\star 2}$	**2.4**$^{\star 3}$	**2.8***	15/15
f_{11}	1000	2200	6300	9800	1.2e4	1.5e4	15/15
0: AdapSS	8.7	5.7	2.6	2.5	2.6	2.7	15/15
1: Bandit	**7.3***	5.1	2.4	**2.3***	**2.4***	**2.5***	15/15
f_{12}	1000	1900	2700	4100	1.2e4	1.4e4	15/15
0: AdapSS	29	18	20	24	12	13	15/15
1: Bandit	27	20	19	20	9.4	**10***	15/15
f_{13}	650	2000	2800	1.9e4	2.4e4	3.0e4	15/15
0: AdapSS	28	13	12	2.6	2.6	2.6	15/15
1: Bandit	**25**$^{\star 2}$	**12**$^{\star 2}$	**11**$^{\star 2}$	**2.4***	2.5	**2.5***	15/15
f_{14}	75	240	300	930	1600	1.6e4	15/15
0: AdapSS	43	34	43	25	20	2.8	15/15
1: Bandit	**33**$^{\star 2}$	**30**$^{\star 2}$	**38**$^{\star 2}$	23	19	2.8	15/15
f_{15}	3.0e4	1.5e5	3.1e5	3.2e5	4.5e5	4.6e5	15/15
0: AdapSS	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	470	∞	∞	∞	∞	∞*2.0e6*	0/15
f_{16}	1400	2.7e4	7.7e4	1.9e5	2.0e5	2.2e5	15/15
0: AdapSS	2.2e4	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
f_{17}	63	1000	4000	3.1e4	5.6e4	8.0e4	15/15
0: AdapSS	23	13	6.6	1.9	1.8	18	7/15
1: Bandit	23	13	6.6	1.9	1.9	5.5	13/15
f_{18}	620	4000	2.0e4	6.8e4	1.3e5	1.5e5	15/15
0: AdapSS	13	5.4	1.9	8.4	6.5	7.8	7/15
1: Bandit	11	**4.6***	1.7	3.2	11	28	5/15
f_{19}	1	1	3.4e5	6.2e6	6.7e6	6.7e6	15/15
0: AdapSS	1.8e3	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	1.3e3	9.5e6	∞	∞	∞	∞*2.0e6*	0/15
f_{20}	82	4.6e4	3.1e6	5.5e6	5.6e6	5.6e6	14/15
0: AdapSS	46	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	40	**640**$^{\star 3}$	∞	∞	∞	∞*2.0e6*	0/15
f_{21}	560	6500	1.4e4	1.5e4	1.6e4	1.8e4	15/15
0: AdapSS	12	350	570	550	520	460	3/15
1: Bandit	12	610	570	550	520	460	3/15
f_{22}	470	5600	2.3e4	2.5e4	2.7e4	1.3e5	12/15
0: AdapSS	1.6e3	2.3e3	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	670	1.4e3	∞	∞	∞	∞*2.0e6*	0/15
f_{23}	3.2	1600	6.7e4	4.9e5	8.1e5	8.4e5	15/15
0: AdapSS	1.5	8.5e3	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	1.5	2.4e3	440	∞	∞	∞*2.0e6*	0/15
f_{24}	1.3e6	7.5e6	5.2e7	5.2e7	5.2e7	5.2e7	3/15
0: AdapSS	∞	∞	∞	∞	∞	∞*2.0e6*	0/15
1: Bandit	∞	∞	∞	∞	∞	∞*2.0e6*	0/15

Table 1: Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009 (given in the respective first row) for different Δf values for functions f_1–f_{24}. The median number of conducted function evaluations is additionally given in *italics*, if $\mathrm{ERT}(10^{-7}) = \infty$. #succ is the number of trials that reached the final target $f_{\mathrm{opt}} + 10^{-8}$. 0: AdapSS is PM-AdapSS-DE and 1: Bandit is F-AUC-Bandit. Bold entries are statistically significantly better compared to the other algorithm, with $p = 0.05$ or $p = 10^{-k}$ where $k > 1$ is the number following the $\star$ symbol, with Bonferroni correction of 48.

out to be completely invariant with relation to monotonous transformations over a given fitness function.

The *F-AUC-Bandit* was compared to yet another adaptive scheme recently proposed to the same objective, the *PM-AdapSS-DE* [7], which uses the Probability Matching technique to select between the strategies, based on the recent relative fitness improvements brought by their respective applications within the current generation.

It is important to remember that the objective of this work was not to compete with the state-of-the art optimizers, but rather to present an adaptive scheme that facilitates the utilization of the algorithm under scrutiny, the Differential Evolution. Besides saving the user from the non-trivial and time-consuming need of defining which strategies should be used for the problem at hand, it automatically adapts the behavior of the optimization process with relation to the different stages of the search. Both analyzed techniques showed to fulfill this aim, while presenting somehow competitive results when compared to the BBOB-2009 candidates.

Although presenting a much higher complexity, in terms of implementation and computational time, the comparison-based credit assignment scheme of the *F-AUC-Bandit* technique showed to be valuable. Significantly better ERT results were achieved by it over the baseline technique on most of the functions, especially in the separable, moderate and ill-conditioned ones, being equivalent otherwise. In the last two classes, multi-modal and weak-structured, almost no difference was found between them, with not so competitive results being attained by both.

Concerning the hyper-parameter configuration of the adaptive schemes, the same parameter values were found to be the best when considering all functions of dimensions 5 and 20. Although these techniques are said to be robust with relation to their hyper-parameters, this could be further checked by repeating the off-line tuning procedure considering, for example, each class of functions individually (thus augmenting the crafting effort).

Another path that could also be explored concerns the use of the diversity measure combined with the fitness for the credit assignment, what has already shown to be very beneficial in the context of multi-modal SAT problems [12, 11]. Besides, an automatic adaptation of other DE parameters could also be considered, such as the F and CR adaptation implemented by the SaDE scheme [15].

7. REFERENCES

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[2] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In W. B. Langdon et al., editor, *Proc. Genetic Evol. Comput. Conf.*, pages 11–18. Morgan Kaufmann, 2002.

[3] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

[4] A. Fialho, M. Schoenauer, and M. Sebag. Toward comparison-based adaptive operator selection. In J. Branke et al., editor, *Proc. Genetic Evol. Comput. Conf.* ACM Press, 2010.

[5] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE, 2009. Updated February 2010.

[6] D. E. Goldberg. Probability matching, the magnitude of reinforcement, and classifier system bidding. *Mach. Learn.*, 5(4):407–425, 1990.

[7] W. Gong, A. Fialho, and Z. Cai. Adaptive strategy selection in differential evolution. In J. Branke et al., editor, *Proc. Genetic Evol. Comput. Conf.* ACM Press, 2010.

[8] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Technical Report RR-7215, INRIA, 2010.

[9] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Technical Report RR-6829, INRIA, 2009. Updated February 2010.

[10] A. W. Iorio and X. Li. Improving the performance and scalability of differential evolution. In *Proc. Conf. Simulated Evol. and Learning*, pages 131–140. Springer-Verlag, 2008.

[11] J. Maturana, A. Fialho, F. Saubion, M. Schoenauer, and M. Sebag. Extreme compass and dynamic multi-armed bandits for adaptive operator selection. In *Proc. IEEE Congress on Evol. Comp.*, pages 365–372. IEEE Press, 2009.

[12] J. Maturana, F. Lardeux, and F. Saubion. Autonomous operator management for evolutionary algorithms. *J. of Heuristics*, 2010.

[13] K. Price. Differential evolution vs. the functions of the second ICEO. In *Proc. IEEE Congress on Evol. Comp.*, pages 153–157, 1997.

[14] K. Price, R. Storn, and J. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*. Springer-Verlag, Berlin, 2005.

[15] A. K. Qin, V. L. Huang, and P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans. on Evol. Comput.*, 13(2):398–417, Apr 2009.

[16] R. Storn and K. Price. Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optim.*, 11(4):341–359, Dec 1997.

[17] D. Thierens. An adaptive pursuit strategy for allocating operator probabilities. In *Proc. Genetic Evol. Comput. Conf.*, pages 1539–1546, 2005.